SUNFEST Technical Report TR-CST01DEC05, Center for Sensor Technologies, Dept of Electrical and Systems Eng, Univ. of Pennsylvania, Philadelphia, PA 2005

University of Pennsylvania

SUNFEST

NSF REU Program
Summer 2005

WORKING TOWARD A BETTER VISION-BASED OBSTACLE DETECTION METHOD

NSF Summer Undergraduate Fellowship in Sensor Technologies
Roman Geykhman (Dept. of Electrical and Systems Engineering) - University of Pennsylvania
Advisor: Dr. Dan Lee

ABSTRACT

Obstacle detection is a vital component of any autonomous mobile robotics application. Vision-based systems for obstacle detection offer the advantage of using relatively inexpensive and readily available video cameras to supply a mobile robot with information about its environment. The key challenge, however, is that cameras supply too much information and complicate the efficient, automated extraction of meaningful features from raw images. Fast and effective obstacle acquisition from this input is still an unsolved problem in robotics.

This paper documents the experimental development of an accurate and efficient vision-based obstacle detection method using the Learning Applied to Ground Robotics (LAGR) experimental platform. The LAGR platform is designed to test algorithms for the successful autonomous navigation of an unmanned vehicle through rural terrain, relying almost entirely on vision to collect information about the environment. The platform is equipped with two pairs of stereo cameras, each with a dedicated Pentium-M computer committed to processing its input and converting it into meaningful information about obstacles present in the platform's environment.

This project focuses on the use of low-level filtering and curve-fitting techniques to enable mobile robots to extract enough information about surrounding obstacles to successfully avoid them. The study is focused on developing and testing new and preexisting algorithms that will also enable the robot to avoid false detection without expending too much computation time in the process. The algorithms will be implemented in C and MATLAB code and tested on board the LAGR platform using both real-time and prerecorded stereo image pairs.

Table of Contents

1. INTRODUCTION

Effective obstacle detection is a vital component of any mobile robotics application. Several methods exist for obtaining information about a robot's environment, using various sensors such as RADAR, SONAR, and laser range-finders. These types of sensors are generally quite accurate, and can be calibrated to provide very precise range and direction data about obstacles in the environment. However, despite any possible accuracy and range benefits these methods offer, they suffer from the fact that they each require specialized and expensive equipment to implement.

Vision-based obstacle detection methods offer the significant advantage of using relatively inexpensive, off-the-shelf video cameras as the chief method for obtaining information about the environment. It also offers the guarantee that each frame of video carries enough information about the environment to sufficiently detect all visible obstacles. Whereas typical SONAR receivers and laser range finders are usually configured to receive data from only one direction or only one plane, cameras are able to see and record a significant portion of the mobile robot's environment and get the “whole picture.” The problem with vision-based obstacle detection methods is that, while they provide sufficient information about obstacles, they in fact provide too much information, making the effective recognition of features a relatively involved process.

Numerous existing techniques can be used to extract meaningful features from images, and to classify those features as obstacles in the robot's environment. This paper will focus on the development of a relatively fast and accurate low-level system to detect obstacles from pairs of stereo images in order to facilitate the autonomous navigation of the Learning Applied to Ground Robotics (LAGR) experimental platform. These techniques and methods will focus mainly on modeling the low-level geometry of the robot's environment, using both commercial and custom-coded vision software.

2. PRIOR WORK IN ROBOT VISION

2.1 Stereo Vision Basics

One of the simpler obstacle detection methods involves the use of fixed pairs of cameras in order to triangulate the locations of obstacles. As Figure 1 illustrates, two images are captured using neighboring cameras pointed at the same object. Each camera has a set of pixel coordinates for each point on this object. The triangle outlined in red lies on the epipolar plane. By definition, this plane passes through the point in space under observation and the optical centers of both cameras. It is evident that corresponding pixels in the two cameras must lie on the same epipolar plane.

It is a fact that this plane does not always intersect horizontal scanlines. However, if the relative orientation and displacement of the two cameras' optical centers is known, a coordinate transformation can be performed on both images in order to map epipolar lines in the raw image to horizontal scanlines in the transformed image. This process is known as rectification and reduces the search for matching image features to a one-

dimensional problem, as matching image features are, by construction, mapped to the same horizontal scanline in both images.

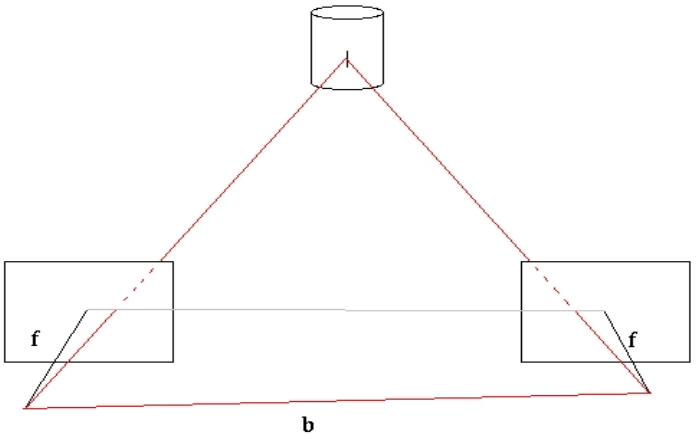


Figure 1: Geometry of Stereo Vision.
Epipolar plane is in red. Epipolar line is in grey.

The triangulation equations are illustrated in Figure 2, and the equations for the real-world coordinates of the object under observation are enumerated in Table 1. By inspection, the equations are derived from manipulations of similar triangles in Figure 2.

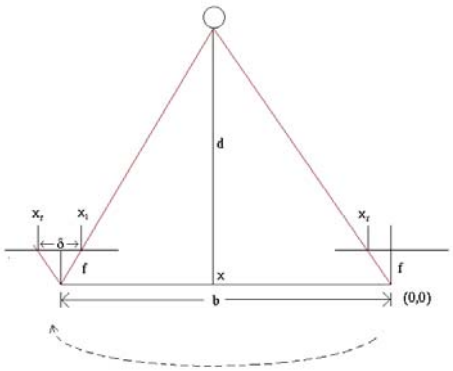


Figure 2: Triangulation Geometry in Stereo Vision

$$\frac{x}{d} = \frac{x_r}{f}$$

$$\frac{b + x}{d} = \frac{x_l}{f} = \frac{x_r + \delta}{f}$$

…

$$d = \frac{fb}{\delta}$$

$$x = \frac{x_r d}{f}$$

$$y = \frac{y_r d}{f}$$

Table 1: Triangulation Equations in Stereo Vision

The equations in Table 1 reveal that in three dimensional space, the x-, y-, and z-coordinates of the image feature are all proportional to the inverse of disparity. Consequently, the uncertainties in these coordinates are proportional to the inverse square of the disparity. This error is nearly negligible for close objects, but for far objects, it can become as high as 1 meter. Given also that at large distances, the quantization of disparity values becomes noticeable, it is not reasonable to expect truly accurate data at low disparities. Subpixel interpolation methods exist to increase the precision of the feature matching algorithms at low disparity, and these algorithms may significantly improve the precision of range data obtained from the stereo image pair [1, pp. 35-38]. But there is an associated computation cost which, for the purpose of rough estimates of obstacle position (the required map resolution is only 0.5 m), is unnecessary.

2.2 The Correspondence Problem

Finding matches between features in stereo image pairs is known as the correspondence problem. The usual methods for establishing the necessary correspondence between image features involve comparing pixels from one image to possible corresponding pixels in the other image and determining a “goodness of match” criteria for the pairing.

A very popular and relatively fast method for establishing correspondence between features in grayscale images is the minimization of the sum of absolute differences of pixel values in patches of image in the left and right frames (as a function of displacement). In this method, a small patch in one image is compared pixel-by-pixel with identically sized patches in the other image which lie along the same scan line. The sum of each difference between corresponding pixels is then taken to be the difference between the two patches. Under ideal conditions, the minimum of this difference would occur where the two patches contain views of the same physical object in their respective images. Once this minimum is found, the coordinate difference between these matching patches is the disparity of that image feature. A pair of rectified images taken from a pair of stereo cameras is shown in Figure 3a. A disparity image, generated from this image pair with 7 x 7 pixel patches, is shown in Figure 3b. Blue indicates low disparity and yellow indicates higher disparity. Dark red indicates no data.

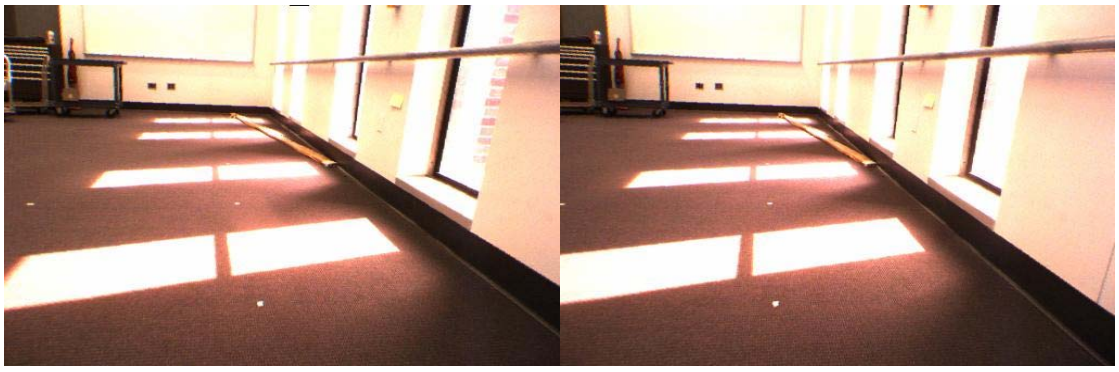


Figure 3a: Left and Right Rectified Images

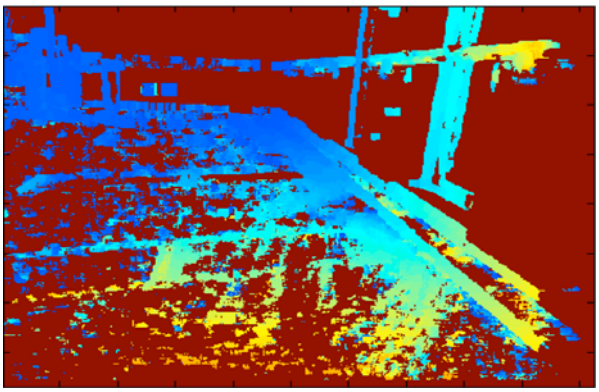


Figure 3b: Disparity Image

Correspondence matching, is, however, not an error-free process. As is evident from Figure 3a and Figure 3b, establishing an accurate correspondence between image features is not a trivial task. The disparity image in Figure 3b does not capture information about the wall to the right side, the far wall with the whiteboard, and parts of the floor. Indeed, it is seen that regions with low contrast are not captured in the disparity image calculation. Such regions offer insufficient features for making a definitive match between image patches by the sum of absolute differences algorithm operating on 7 x 7 pixel neighborhoods. This effect is illustrated in Figure 4.

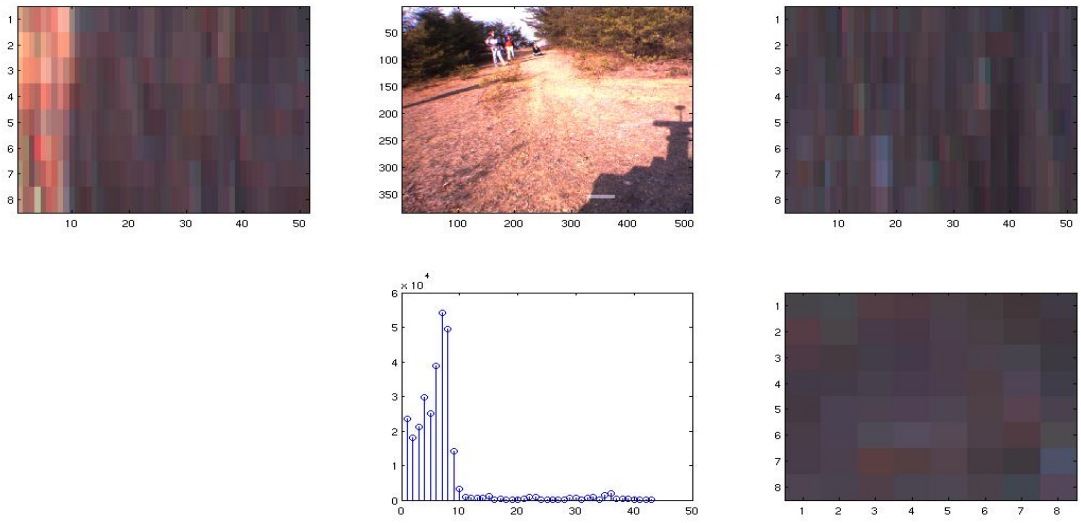


Figure 4: Sum of Absolute Differences Algorithm in Low Contrast Regions

The center top pane shows a rectified image with the region of comparison highlighted in the bottom right of the image. The top left and top right panes show a magnification of the region of interest in the left and right rectified images. The bottom right pane shows the 8 x 8 pixel magnification of the patch in the right image being compared with successive patches in the left image in the disparity calculation. The bottom center pane shows the sum of absolute differences as a function of displacement. Now, the region of interest lies on the ground directly in front of the robot, approximately

1 meter in front of the camera. With the geometry of the camera system, this would correspond to a disparity value of approximately 40 pixels. However, as can be seen in the bottom center pane, the low contrast of the region of interest causes very low sum of absolute difference values for displacements of anywhere from 10 to 45 pixels. Indeed, with this kind of difference data, no clear minimum is evident and a disparity value cannot be assigned without ambiguity. In fact, it is entirely plausible that random errors in the images determine the absolute minimum of this difference, and that the horizontal coordinate of this minimum value will have no relation to the actual disparity of the image feature being observed. The identification and elimination of these kinds of uncertain disparities is known as validation, and will be discussed in Section 4.

2.3 A Review of Obstacle Detection Methods

In order to identify obstacles in an image, it is necessary to perform some kind of segmentation. In one way or another, pixels (and their projections into 3D space) need to be grouped into obstacles and non obstacles. The search for effective segmentation processes is a topic of ongoing research in computer vision, and various methods and criteria are being studied to identify whole objects in images. Despite the precise detail and robustness that these methods promise, they are too computationally intensive for the current hardware to execute in a reasonable time frame.

Segmentation methods such as the normalized cuts approach [2] yield very good segmentation results on complex scenes. However, they take several minutes just to segment a single frame of video. While these techniques can be a springboard toward more complicated tasks such as higher level object extraction, recognition, and classification, for the purpose of simply *detecting* obstacles, they do too much work.

The precision of hi-level segmentation notwithstanding, low-level techniques can accomplish quite a bit in the area of simple detection. The majority of these methods work with disparity images generated from rectified pairs of stereo images. Successful low-level operations are possible on this kind of data, as it contains all the information about the 3D coordinates of every pixel in the camera image.

Many low-level obstacle detection techniques involve the brute-force projection of a three-dimensional point cloud into the robot's environment. Several possibilities exist for further processing the data. Some of the more successful techniques are outlined below.

One approach, as described in [3], is to take the point cloud and extract, by various statistical methods, an estimate for the ground plane. This estimate is then converted into a set expected disparity values for every point in the image. Significant deviations from this expected value are classified as obstacles, and the coordinates of each pixel comprising these obstacles are then easily calculated and projected into a map of the environment.

The current implementation of the obstacle detection algorithm on the LAGR platform uses a similar method to populate a cost map of the environment with counts of stereo points that lie above a similarly-obtained ground plane estimate. This cost map is then taken directly to the planning algorithm.

Noise due to false matches can be a big problem with both of these methods. The process of using raw stereo count data to populate a local obstacle or cost map is very susceptible to false detections brought about by spurious data.

The other, more critical problem arises from false correspondences with high disparity. This noise will generally be projected very close to the robot. Furthermore, given the projective nature of the camera, it will all be projected into the same region near the front of the camera. This will artificially inflate the stereo point count of the region directly in front of the robot and cause an artificial obstacle to appear directly in the robot's path. For purposes of navigation, this poses a significant problem. In practice, a certain region in front of the robot is always assumed to be obstacle-free and stereo data that places obstacles in it is ignored. This eliminates problems caused by false matches, but also makes it impossible to detect actual obstacles in that region.

D.R. Murray's work in stereo vision [1] brings up several key problems in the filtration and processing of disparity images. One issue is that the projective nature of the camera reduces the pixel count of objects as their distance from the camera increases, thereby reducing the number of data points for far-off objects. The other is that manipulating raw point data is subject to certain limitations. As has been mentioned, manipulations of this data are highly susceptible to noise, lose accuracy with distance, and effectively do too little work to detect obstacles.

Furthermore, low level noise reduction algorithms can only go so far. It is a self-evident fact that humans have a certain intuition about vision that allows them to make high-level descriptions of complex scenes without necessarily requiring access to the low level computations that occur in their visual cortex. One of these intuitive notions is the fact that most obstacles that may be encountered in the world are contiguous elements in three-dimensional space, and generally are smooth and thus exhibit only gradual changes in the surface normal vectors over their surfaces.

Murray's work has shown that surface orientation is indeed a good criterion to use in segmenting stereo images for use in higher-level computations and algorithms. The remainder of this paper will explore the use of surface orientation as a low-level spring-board to accurate and fast obstacle detection.

3. HARDWARE AND DESIGN REQUIREMENTS

The purpose of this project was to develop an improved vision-based obstacle detection algorithm for the Learning Applied to Ground Robotics (LAGR) platform, shown in Figure 5. This platform is equipped with two pairs of stereo cameras, a dedicated 2GHz Pentium M computer for each pair, a central planning computer, and a low level computer to interface with the robot's hardware.

Figure 5: LAGR Test Platform

3.1 Onboard Cameras and Vision Software

The two onboard camera pairs come with the commercial Triclops/Digiclops stereo matching and image rectification software. This software includes several built-in filters and validation methods. Stereo point correspondence is established by a Sum of Absolute Differences algorithm. Several validation methods are used to eliminate false matches: a texture check, which ensures that featureless, low-contrast, regions are not scanned, a uniqueness check, which checks the "goodness" of the minimum in the sum of absolute differences compared with other minima along the scanline, and a surface size validation check. The exact details of the algorithms implemented in the Triclops libraries are not made available for inspection. Their effectiveness is discussed in Section 4.2. The image resolution used for this project was typically 512 x 384 pixels.

3.2 Design Goal

The LAGR platform is tested in a rural environment, and is expected to be able to navigate its way around reasonably textured obstacles such as trees, bushes, shrubs, rocks, and fences. It is not required to classify these objects as anything other than obstacles in order to navigate around them. The robot will be navigating mostly over dirt trails and grassy surfaces, which experience has shown, generally provide sufficient texture to be accurately recognized as contiguous surfaces for ground detection. Typical examples of the terrain are shown in Figure 6.

Figure 6: Typical Terrain for Autonomous Navigation

The primary need is to design a software system that will accurately detect obstacles observed through the pairs of stereo cameras and construct a map of the robot's environment as faithfully and reliably as possible, while utilizing as little computation time as possible. A desired value for processing time of one frame of video is typically less than half a second.

4. DEVELOPMENT OF A BETTER VISION ALGORITHM

4.1 Initial Vision System Status

Initially, obstacle detection on the LAGR platform was done using the projection of a 3D point cloud from a filtered disparity image. This point cloud was then broken up into a two-dimensional grid of rectangular columnar cells 0.5 m in width by 0.5 m in length. Cells in which the vertical standard deviation of stereo points was low were used in order to extract a ground plane estimate. This ground plane estimate was then used to classify the remaining stereo points as obstacles if they fell in a region from approximately 0.25 to 1.25 meters above the ground plane. A traversal cost was then calculated for each cell based on the number of stereo points in each cell classified as obstacle points. This cost was taken directly as the obstacle map. An example of such a map, constructed from an indoor scene, is shown in Figure 7. Lighter cells indicate high traversal cost and darker cells indicate lower traversal cost.

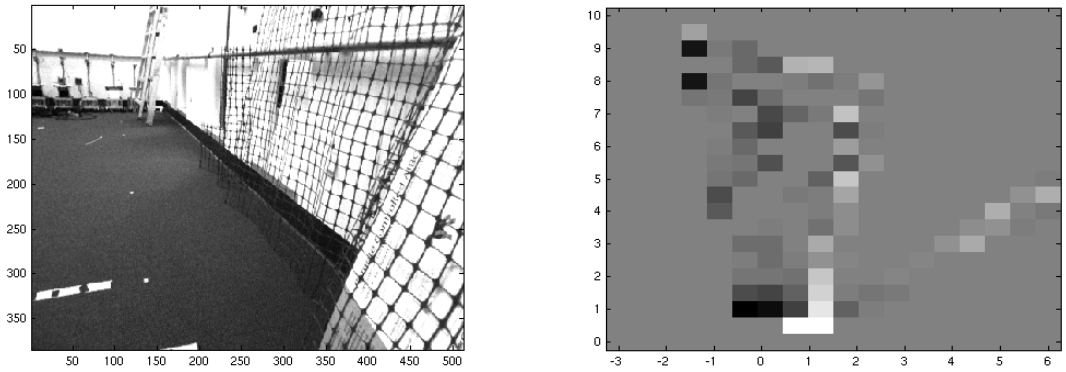


Figure 7: Indoor Scene and Cost Map Generated By Point Cloud Projection

This approach suffers from one main drawback. If a cell contains both ground points and obstacle points, it will not be used in the ground plane estimate because of the high spread in the z-coordinates of the points in the cell. Furthermore, the criterion used to classify a cell as containing ground points assumes that the ground is relatively flat in the robot coordinate system. As Figure 8 shows, a flat sloping ground surface will yield a higher vertical standard deviation, and will be more likely to be rejected for ground plane estimation. Furthermore, if the robot is facing a sloping hill, it may not get any cells with sufficiently low z-coordinate standard deviation, default to using a flat-ground assumption, and register the hill's stereo points as an obstacle, impeding the robot's ability to effectively reach its goal.

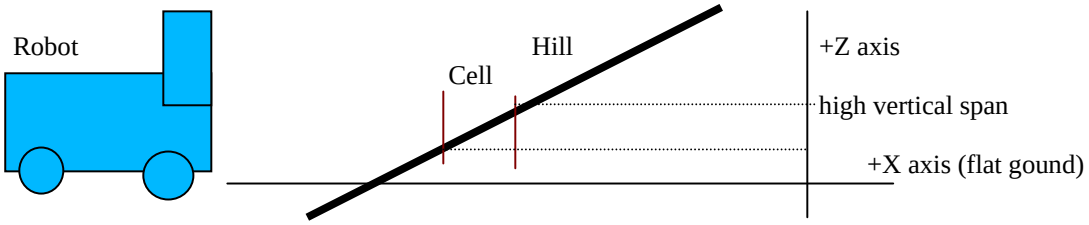


Figure 8: Problems Caused by Sloping Ground

The solution to this problem is to increase the threshold for the maximum vertical standard deviation a cell can contain to still be considered a valid ground point. Unfortunately, this will have the detrimental effect of throwing off the accuracy of the ground plane estimate. This ground plane issue, has, in fact, been a major problem in the vision system to date.

The other problem with the vision system has been the false detections caused by the incorrect matches discussed in Section 2. Specifically, incorrect matches in low-contrast regions such as the sky have resulted in false obstacles being detected directly in front of the robot, causing problems with navigation.

4.2 Initial Attempts at Improvement of the Vision System

The first attempt to improve the vision system involved experimentation with filtration and validation methods on the stereo matching algorithm. Various methods for rejecting false matches generated by the sum of absolute differences minimization were explored.

Three of these algorithms were supplied with the Triclops package that came with the LAGR platform. These were the texture validation, uniqueness validation, and surface size validation. The fourth algorithm was a back-checking confirmation, which compared left-to-right disparity values with their right-to-left counterparts. Mismatches were rejected as invalid.

The problem with the texture and uniqueness validation was that while they succeeded in eliminating most of the spurious data, they also eliminated valid obstacle and ground points. Because the obstacle detection algorithm relies heavily on being able to extract an accurate ground plane estimate from the stereo data, this posed a significant hindrance. After some experimentation on prerecorded images, it was decided that any benefit of error reduction was outweighed by the cost of loosing valuable ground and obstacle data, and consequently, these two validation methods were not employed.

The back-checking confirmation algorithm proved quite effective at eliminating false readings without rejecting valid readings. The disparity images produced by this algorithm were generally accurate and had few incidences of false data, especially when combined with relatively mild uniqueness and texture checks. A significant problem with this approach was the additional overhead and computation time required to make the comparison for every pixel in the disparity image. In practice, the generation of the disparity image takes about 33% of the total computation time for each frame of video,

and the cost of increasing it by executing what amounted to only a low-level filtration step was not acceptable.

The final validation technique, surface size validation, proved extremely effective at eliminating false detections while preserving true readings. This method works on the assumption that spurious readings will be small in size in the disparity image and rejects objects that fall below a certain size threshold. This method was so effective that it was incorporated into all subsequent obstacle detection algorithms.

Yet despite any improvements gained from low level filtration techniques, spurious readings still managed to find their way into the disparity image, and the problems of sloping ground and cells occupied by both ground and obstacles remained. These problems served as the motivation for a higher level analysis of the disparity image.

4.3 Obstacle Detection Based on Disparity Image Segmentation

A cursory glance at any disparity image, such as Figure 9, will immediately suggest a method of obstacle detection. It is a fact that obstacles appear in the disparity image as large contiguous objects with gradually changing disparity values over their area. It was thought that the extraction of such large contiguous objects could be an effective method of obstacle detection.

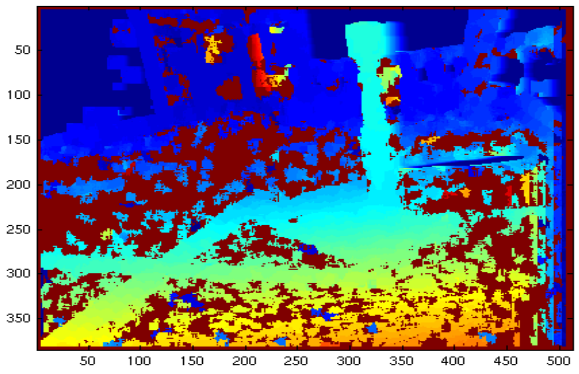


Figure 9: A Disparity Image

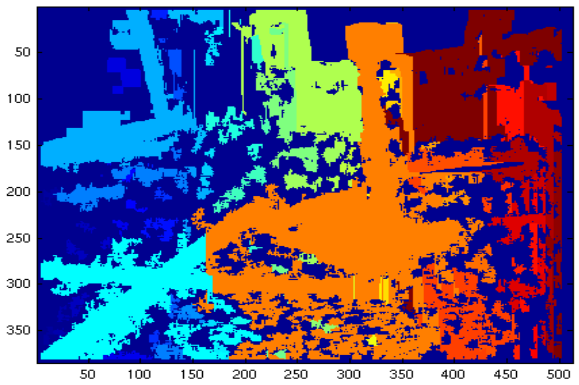


Figure 10: Segmentation of Figure 9. Regions of the same color are unified objects.

A simple union find algorithm—similiar to the one used to measure the size of contiguous objects for the surface size validation—was run on disparity images captured through the LAGR's stereo cameras. As Figure 10 shows, many obstacles seen in the disparity image in Figure 9 are indeed successfully segmented out and separated into discrete units. Once this is done, criteria such as real world size and height above the ground may be used to populate not a continuous-valued cost map susceptible to false readings, but a discrete-valued binary map showing precisely where the large obstacles are in the robot's environment.

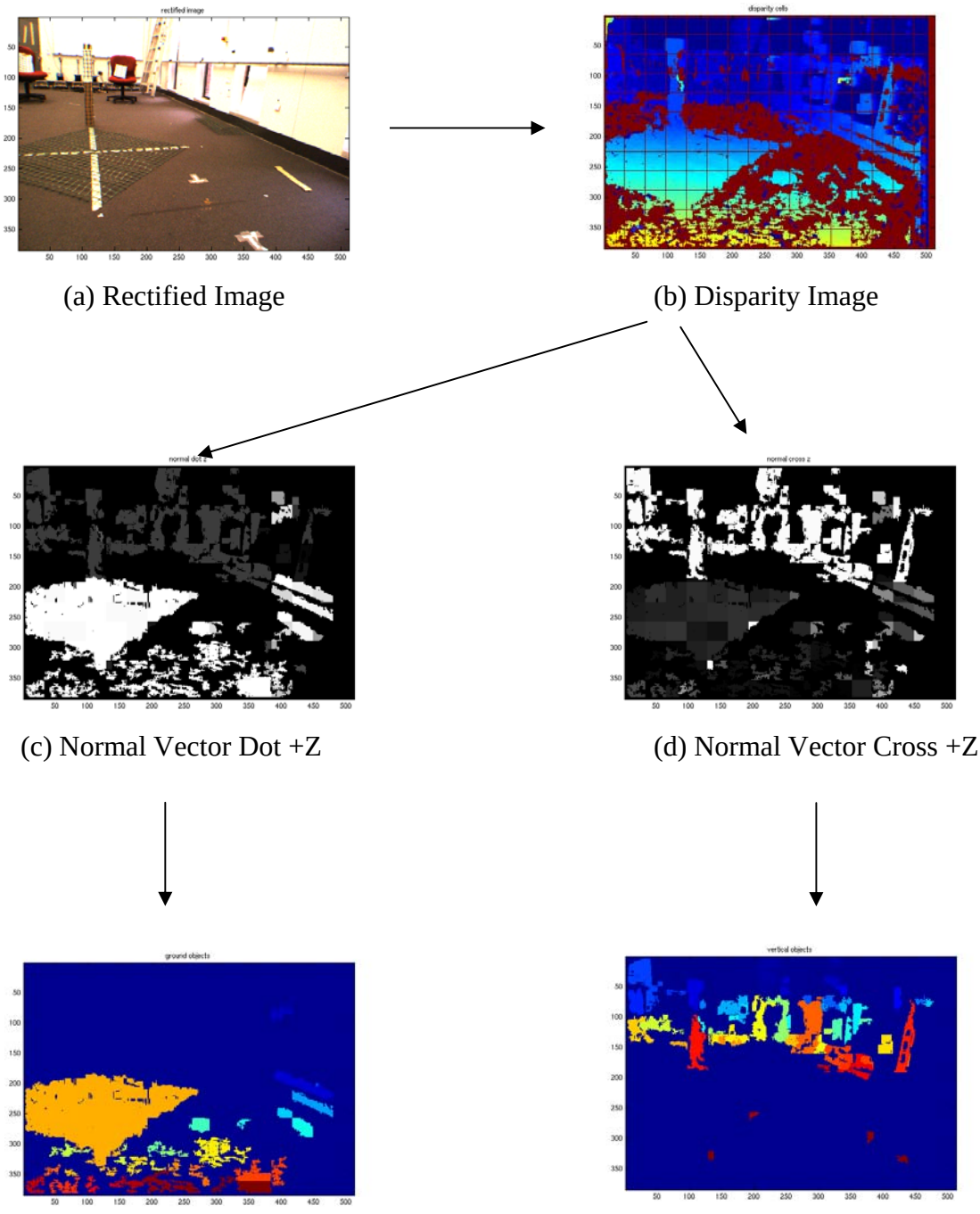
However, this simplistic approach suffers from two related and important drawbacks. First, it does not solve the problem of obtaining an accurate ground plane estimate, which, as was mentioned before, creates a problem in sloping or uneven terrain. Second, as is evident in Figure 10, objects that are sitting on the ground blend into the ground in the disparity image, and a union-find algorithm will classify both the object and the ground it is sitting on as a single unit. Potentially, entire swaths of ground and objects will be classified as a unified body, and cause the map to be populated with obstacles where none exist.

What was required was a way to separate the objects that sit on the ground from the ground itself. Murray accomplished this by computing a surface normal vector for a curve fit to each pixel's neighborhood and using it as the segmentation criterion. As seen in [1], this method is quite successful. Unfortunately, it requires quite a bit of computation time—up to 2 minutes per frame for a 320 x 240 image. Even with Murray's proposed code optimizations, his method is designed to extract highly precise information about complicated surfaces. This was not the goal of this design. Since the only required data was a fairly rough estimate of where an obstacle is located, and not any information about its structure, a slightly cruder algorithm was in order.

4.4 The Final Obstacle Detection Algorithm

Now, intuitively, visual information contains only data about the surfaces of nearby objects. Further intuitive reasoning will reveal that most obstacles are vertical and that the ground is mostly horizontal. The final version of the obstacle detection algorithm developed over the course of this project employs these simple observations in order to classify stereo points generated from a regular disparity image as obstacles or ground points based on the orientation of the normal vector of dynamically determined surface elements in the image. A flowchart of the algorithm is given in Figure 11.

First, a disparity image is generated by the commercial Triclops software from rectified image pairs. The right rectified image is shown in 11 (a) and the corresponding disparity image in 11 (b). From (b), the distance image is calculated using the equations of Table 1. In order to save computation time, the x- and y-coordinates are also calculated for each pixel, as they will be required in future steps. Next, the distance image is split into 32 x 32 pixel nonoverlapping regions and the union-find algorithm is executed on these 32 x 32 pixel patches in order to extract contiguous surfaces for the calculation of surface normal vectors. It should be noted that the initial splitting of the image avoids the problem of having objects blend into the ground by keeping the segmentation algorithm confined to the local 32 x 32 pixel region.



(a) Rectified Image

(b) Disparity Image

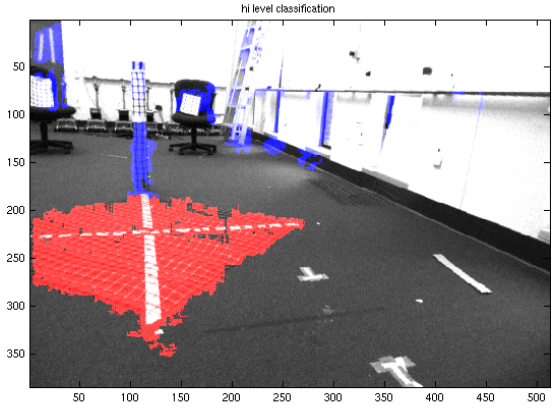
(c) Normal Vector Dot +Z

(d) Normal Vector Cross +Z

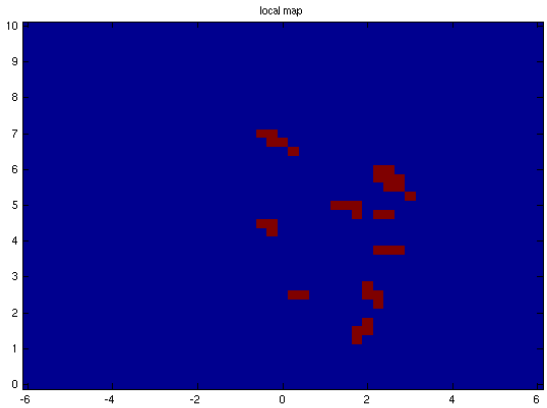
(e) Horizontal Surfaces

(f) Vertical Surfaces

Figure 11: Obstacle Detection Algorithm Flowchart (Part I)



(g) Final Classification of Stereo Points (Red = Ground, Blue = Obstacle)



(h) Bird's Eye Map of Local Environment (Robot is at (0,0); Red Dot = Obstacle)

Figure 11: Obstacle Detection Flow Chart (Part II)

For each contiguous object extracted by the union find algorithm in the 32 x 32 pixel patches, the 3D coordinates of all the pixels comprising that object are collected. From these points, a plane of best fit is calculated by means of orthogonal distance regression. As Figure 12 and the equations in Table 2 show, this problem is solved by finding the eigenvectors of the covariance matrix generated by the 3D coordinates of the points comprising the object. The eigenvector corresponding to the minimum eigenvalue gives the direction of the surface normal vector.

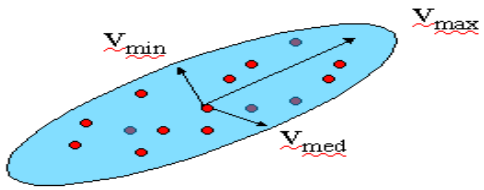


Figure 12: Orthogonal Distance Regression

$$\begin{bmatrix} \frac{1}{n}\sum dxdx & \frac{1}{n}\sum dxdy & \frac{1}{n}\sum dxdz \\ \frac{1}{n}\sum dydx & \frac{1}{n}\sum dydy & \frac{1}{n}\sum dydz \\ \frac{1}{n}\sum dzdx & \frac{1}{n}\sum dzdy & \frac{1}{n}\sum dzdz \end{bmatrix} = \begin{bmatrix} \vec{v}_1 & \vec{v}_2 & \vec{v}_3 \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \cdot \begin{bmatrix} \vec{v}_1 \\ \vec{v}_2 \\ \vec{v}_3 \end{bmatrix}$$

Table 2: Equations for Orthogonal Distance Regression

Once this surface normal vector is computed for each object in each 32 x 32 pixel patch, it is dotted with the +Z direction of the vehicle coordinate frame. Absolute values of this dot product near 1.0 indicate a highly horizontal surface likely to be the ground and absolute values of this dot product near 0.0 indicate vertical surfaces most likely to be obstacles. This dot product value is recorded in a new image for each pixel in each contiguous object, and the corresponding cross product value (= sqrt($1 - \text{dot}^2$)) is also recorded in a separate image. Figure 11 (c) and 11 (d) show the absolute value of the dot and cross products of the surface normal vectors with the +Z axis in the vehicle coordinate frame. It can be seen that in Figure 11 (c), ground points are brighter, representing a high value for the dot product, and in 11 (d), the cross product values are higher for vertical obstacles.

After this initial low level classification step, a ground plane estimate is obtained by searching for large contiguous regions of pixels classified as horizontal ground points with a union-find algorithm operating only on pixels belonging to sufficiently horizontal surfaces. In Figure 11, orientations within 30 degrees of absolutely horizontal were included. The results of the union-find are then processed by an arbitrarily-chosen plane-fitting algorithm. For the purposes of this experiment, orthogonal distance regression was used in this step. Similarly, vertical obstacles are extracted by running the same union find algorithm on the distance image, but now using only the pixels previously classified as belonging to vertical surfaces by the low level preliminary classification step. The horizontal and vertical results of the union-find algorithm are displaying in Figure 11 (e) and 11 (f), respectively.

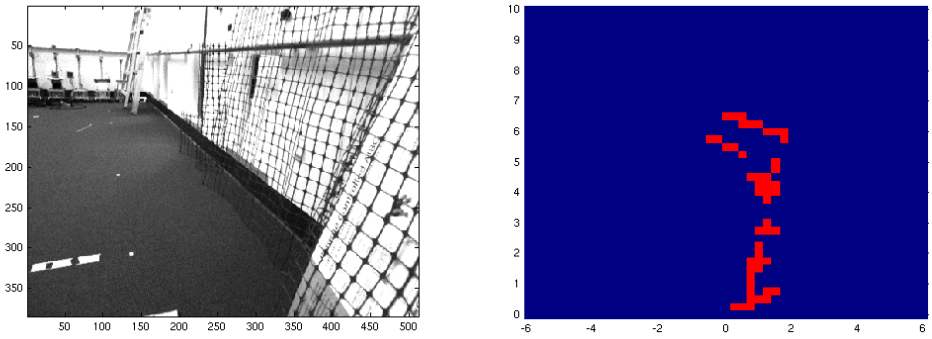
Once the ground plane estimate is obtained, and the coordinates of the vertical obstacles are extracted from the step above, the map generation step is trivial. Each vertical obstacle has a set of points with coordinates in the vehicle coordinate system. The point has a vertical span, a horizontal span, and a horizontal orientation given by the *maximum* eigenvector of the covariance matrix generated by the x- and y-coordinates of the points corresponding to the obstacle. These quantities are compared with certain thresholds for height above the ground plane, pixel count, and absolute size in the robot coordinate system. Objects satisfying these thresholds are classified as valid obstacles. Figure 11 (g) shows the final classification of pixels in the stereo image superimposed onto the rectified image. Areas shaded red are composed of pixels classified as belonging to large horizontal objects assumed to be the ground, and areas shaded blue are comprised of pixels classified as belonging to vertically-oriented objects satisfying the aforementioned criteria. Once these ground points and obstacles are classified, their coordinates are recorded in the local environment map, as shown in Figure 11 (h).

5. SUMMARY OF EXPERIMENTAL RESULTS

The results in Figure 11 and Figure 13 are typical examples of the accuracy with which the algorithm can operate. As can be seen in Figure 11, all major vertical obstacles that have sufficient contrast to be detected by the stereo system are faithfully placed into the local map in Figure 11 (h). Figure 13 (a) shows the indoor scene from Figure 7, with artificially added texture in the form of garden fencing along the otherwise featureless wall along the right side. It is seen in Figure 13 (b) that the algorithm accurately records the entirety of the wall as an obstacle, as well as the ladder and small robots toward the back of the room.

Compared with the results of the current vision algorithm (in Figure 7), the results of the new algorithm in Figure 13 seem much sharper in terms of what is an obstacle in the robot's environment. The entire wall can be seen as impassible, whereas Figure 7 shows fading traversal costs over its length.

The discontinuities in the wall seen at farther distances in the map are a consequence of the uncertainty of obstacle position increasing with lower disparity values, as discussed earlier in Section 2.1. Average execution time per frame of video peaked at 1/6 – 1/5 seconds per frame.



(a) Rectified Image (b) Local Map (Red = Obstacle)

Figure 13: Indoor Scene and Map

6. DISCUSSION AND CONCLUSIONS

Effective obstacle detection is a vital component of autonomous navigation, and vision-based obstacle detection methods offer the advantage of cheap sensors, wide viewing angles, and a guarantee that the raw data from the cameras contains sufficient information about the environment in order to generate an accurate and reliable map. Stereo vision offers the ability to effectively locate objects in three-dimensional space with relatively simple and fast algorithms, and the ability to access object properties such as size, location, and surface orientation.

The algorithm developed over the course of this project uses these simple properties, surface orientation, in particular, to generate accurate, reliable, and stable maps, and has proved to be robust when faced with variable environmental conditions such as sloping terrain. Testing has shown it to be reasonably resistant to noisy stereo data and false detections, as compared to the obstacle detection algorithm previously implemented on the LAGR platform, and as seen in Section 7, the maps it generates are much more decisive in terms of what is and what is not traversable. Furthermore, it is sufficiently fast to be used in real time on a mobile platform. Its success is further confirmation of the effectiveness of using surface orientation as a classifying criterion for 3D stereo data.

7. RECOMMENDATIONS

After considerations of various low-level obstacle detection algorithms, it is recommended that the algorithm developed during the course of this project be implemented on the LAGR platform in order to improve its ability of effectively identify obstacles in its environment. This algorithm offers the advantages of reliability, robustness, and accuracy. Furthermore, the algorithm can be easily implemented as a starting point for more sophisticated obstacle detection techniques that rely on learning algorithms and other hi-level techniques.

8. ACKNOWLEDGEMENTS

I would like to thank the many people without whose generous support, this project would not have been possible. Professor Daniel Lee of the University of Pennsylvania Electrical and Systems Engineering Department has been a source of invaluable wisdom and guidance. His support as an advisor is much appreciated. Paul Vernaza of the University of Pennsylvania GRASP Laboratory is a master guru of robotics hardware. His assistance with technical issues and software allowed this project to proceed smoothly and efficiently. Many thanks also to the National Science Foundation, whose support for undergraduate research initiatives made this project possible, and to the University of Pennsylvania Department of Electrical and Systems Engineering Department for hosting the REU. And, of course, many thanks to Prof. Jan Van der Spiegel for capably managing this undergraduate research program.

9. REFERENCES

1. D. R. Murray, Patchlets: a method for interpreting correlation stereo 3D data, *PhD Thesis, University of British Columbia, 2004.*

2. J. Shi, and J. Malik, Normalized cuts and image segmentation, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol 22, Issue 8, Aug. 2000, 888 – 905.

3.N. Molton, S. Se, J.M. Brady, D. Lee, and P. Probert, A Stereo Vision-Based Aid for the Visually-Impaired, *Image and Vision Computing Journal*, Vol 16, No 4, (1998) 251 - 263.